IN THE SPECIFICATION

Following is a marked-up version of each amended paragraph of the subject patent application. The Examiner is requested to delete the indicated paragraph and replace it with the amended paragraph. The location for each amended paragraph is indicated.

The paragraph on page 1, beginning at line 18, and ending on page 1 at line 32, should be amended as follows:

A typical communication system includes a number of devices or nodes that communicate over a plurality of connections. The system is organized into a plurality of local connections with a limited number of nodes associated with (connected to) each local connection. A network of bridges interconnects the local connections so that each device can communicate with other devices not associated with the same local connection. The bridge for each local connection monitors input traffic from other bridges in the network to determine if traffic originating at another bridge is addressed to a node ~~on~~ connected to it locally. In response, the bridge provides a path that allows the information to pass through to the local connection. Similarly, when information is sourced from the local connection to an external destination node, the bridge allows the information to pass from the local connection to the next bridge on the path to the destination node.

The paragraph on page 7, beginning at line 26, and ending on page 8 at line 7, should be amended as follows:

As shown in Figure 2, a processor 50 communicates bidirectionally with memories 52 and 54, where the instructions representing one or more tree levels are stored. For example, tree levels one (referred to as the root level), two and three of Figure 1 are stored in the memory 52 and tree levels four and five are stored in the memory 54. If the memory 52 has a faster memory access time than the memory 54, then the instructions stored in memory 52 can be accessed faster than those in memory 54. It is known ~~know~~ that a significant number of tree searches are terminated in the root memory or within one or two levels of the root memory. Simulation and analysis show that about 30% of the tree instructions are matched in the root tree memory. Thus if the tree root is stored in the memory 52, the process will likely converge faster.

The paragraph on page 10, beginning at line 16, and ending on page 10 at line 21, should be amended as follows:

In yet another embodiment, it may be possible to store especially critical or frequently-used small trees entirely within the internal memory element 8060. Thus providing especially rapid tree searches for any tree that is located entirely on-chip. The segregation between the tree levels stored within the internal memory 8060 and the external memory 8464 can also be made on the basis of the probabilities of certain patterns in the input data.

Delete the material beginning on page 3, line 9, and ending on page 5, line 17, as follows:

~~One popular prior art searching scheme organizes the search space as a tree having a root portion where the search begins, intermediate branches, and finally a plurality of leaves, where the final decisions or matches occur. An exemplary tree structure is illustrated in Figure 1, where entries or decision points are interconnected by branches. An instruction or bit pattern resides at each decision point for analyzing the input bit pattern (also referred to as the search object) and in response thereto for sending the bit pattern to the next appropriate decision point.~~

~~The tree of Figure 1 has five levels of analysis or entries, as represented by the five columns of vertically aligned nodes, represented by circles. At a start step 12, a five character word or symbol (the search object) is input to the decision tree, for instance the characters AFGZ3. The most significant characters of the symbol are compared with the tree entries at a decision point 14, and the analysis proceeds along a branch 16, representing the symbol "A," to a decision point 18. From there, the process proceeds as follows: branch 20, decision point 22, branch 24, decision point 26, branch 28, decision point 30 and finally branch 32, which is commonly referred to as a leaf of the tree. At this leaf, the symbols have been decoded and the appropriate action or decision associated with that leaf is executed.~~

~~It is known to those skilled in the art that the decision process at each entry of the tree is executed by using a processing device to compare a first number of symbols at the first entry with a first number of the input symbols. The result of the first comparison determines the next branch that the process will follow. The symbols at the second entry are fetched by the processor and a second group of the input symbols are compared with the symbols at the~~

second entry. These alternating fetching and comparing steps are executed as the search object is processed through the tree until a decision entry is reached.

The decision tree, such as that of Figure 1, is stored in a program memory. Each node of the tree is an instruction and the fields of each instruction specify the branches (links) of the tree to which the processor is directed, based on the results of the instruction. The process then moves along the indicated branch to the next tree node. Special instructions are included at the leaves, as these represent the end of the decision process and therefore command a specific action at each leaf.

With reference to Figure 1 the root of the tree is the first node, node 14. Assume the instruction for node 14 is stored at address 0 in the program memory. Further assume that this instruction maps each letter of the alphabet to a node (i.e., there are twenty-six branches from this instruction or node 14). To find the next node for the process the memory address of the current node (zero in this example) is added to the address offset associated with the matching character. Assume the offset address for the letter A is 10, for B the offset address is 11, for C the offset address is 12, etc. Thus node 18 of Figure 1 is located at memory address 10 (base address of zero plus A offset address of 10). The path for the letter B leads to memory address 11 and the path for letter C leads to memory location 12. Since the example of Figure 1 includes only A, B and C as possible matching values, all other letters of the alphabet (D through Z) that are input to the node 14 are directed to memory locations 13 to 35, each containing a leaf instruction that indicates that the input pattern did not match any location in the tree.

Since the input object begins with an A, the process is directed to the node 18 (memory location 10), which contains three instructions or three potential pattern matches and a memory address offset associated with each. If the pattern match is D and the offset address for the D branch is 1, the process moves to the memory location 11 or node 19 in Figure 1. If pattern match is an E and the offset address for the E is 2, the process moves to memory location 12. If pattern match is an F and the offset address for the F is 3, the process moves to memory location 12, or the node 22 via the link 20. If there is no match at this instruction or node, then the process is directed to an offset address of 4, or node 23, where a

~~leaf with a special instruction is located. For the input symbol presented in Figure 1 (i.e., AF), the process moves to the node 22 (memory location 13) via the link 20.~~

Insert the following material after line 20 on page 7:

One popular prior art searching scheme organizes the search space as a tree having a root portion where the search begins, intermediate branches, and finally a plurality of leaves, where the final decisions or matches occur. An exemplary tree structure is illustrated in Figure 1, where entries or decision points are interconnected by branches. An instruction or bit pattern resides at each decision point for analyzing the input bit pattern (also referred to as the search object) and in response thereto for sending the bit pattern to the next appropriate decision point.

The tree of Figure 1 has five levels of analysis or entries, as represented by the five columns of vertically aligned nodes, represented by circles. At a start step 12, a five character word or symbol (the search object) is input to the decision tree, for instance the characters AFGZ3. The most significant characters of the symbol are compared with the tree entries at a decision point 14, and the analysis proceeds along a branch 16, representing the symbol "A," to a decision point 18. From there, the process proceeds as follows: branch 20, decision point 22, branch 24, decision point 26, branch 28, decision point 30 and finally branch 32, which is commonly referred to as a leaf of the tree. At this leaf, the symbols have been decoded and the appropriate action or decision associated with that leaf is executed.

It is known to those skilled in the art that the decision process at each entry of the tree is executed by using a processing device to compare a first number of symbols at the first entry with a first number of the input symbols. The result of the first comparison determines the next branch that the process will follow. The symbols at the second entry are fetched by the processor and a second group of the input symbols are compared with the symbols at the second entry. These alternating fetching and comparing steps are executed as the search object is processed through the tree until a decision entry is reached.

The decision tree, such as that of Figure 1, is stored in a program memory. Each node of the tree is an instruction and the fields of each instruction specify the branches (links) of the tree to which the processor is directed, based on the results of the instruction. The

process then moves along the indicated branch to the next tree node. Special instructions are included at the leaves, as these represent the end of the decision process and therefore command a specific action at each leaf.

With reference to Figure 1 the root of the tree is the first node, node 14. Assume the instruction for node 14 is stored at address 0 in the program memory. Further assume that this instruction maps each letter of the alphabet to a node (i.e., there are twenty-six branches from this instruction or node 14). To find the next node for the process the memory address of the current node (zero in this example) is added to the address offset associated with the matching character. Assume the offset address for the letter A is 10, for B the offset address is 11, for C the offset address is 12, etc. Thus node 18 of Figure 1 is located at memory address 10 (base address of zero plus A offset address of 10). The path for the letter B leads to memory address 11 and the path for letter C leads to memory location 12. Since the example of Figure 1 includes only A, B and C as possible matching values, all other letters of the alphabet (D through Z) that are input to the node 14 are directed to memory locations 13 to 35, each containing a leaf instruction that indicates that the input pattern did not match any location in the tree.

Since the input object begins with an A, the process is directed to the node 18 (memory location 10), which contains three instructions or three potential pattern matches and a memory address offset associated with each. If the pattern match is D and the offset address for the D branch is 1, the process moves to the memory location 11 or node 19 in Figure 1. If pattern match is an E and the offset address for the E is 2, the process moves to memory location 12. If pattern match is an F and the offset address for the F is 3, the process moves to memory location 12, or the node 22 via the link 20. If there is no match at this instruction or node, then the process is directed to an offset address of 4, or node 23, where a leaf with a special instruction is located. For the input symbol presented in Figure 1 (i.e., AF), the process moves to the node 22 (memory location 13) via the link 20.